EEEEEEEEEEEEE	RRRRRRRRRRRR	FFFFFFFFFFFFF
EEEEEEEEEEEEE	RRRRRRRRRRR	FFFFFFFFFFFFF
ÉÉÉÉÉÉÉÉÉÉÉÉÉÉ	RRRRRRRRRRR	FFFFFFFFFFFFF
EEE	RRR RRR	FFF
EEE		
	RRR RRR	FFF
EEE	RRR RRR	FFF
EEE	RRR RRR	FFF
EEE	RRR RRR	FFF
ĒĒĒ	RRR RRR	FFF
EEEEEEEEEE	RRRRRRRRRRR	FFFFFFFFFF
EEEEEEEEEEE	RRRRRRRRRRRR	FFFFFFFFFF
EEEEEEEEEE	RRRRRRRRRRRR	FFFFFFFFFF
EEE	RRR RRR	FFF

EEE	RRR RRR	FFF
EEEEEEEEEEEE	RRR RRR	FFF
EEEEEEEEEEEEE	RRR RRR	FFF
EEEEEEEEEEEE	RRR RRR	FFF
	mm nm	111

RRRRRRR RRRRRRR RR RR RR RR RR RR RR RR RRRRRR	XX	DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD	\$	KK
		\$		

RXC

PRO

ENT

0

VAR

SUBROUTINE RXDISK (LUN)

C Version:

(*

(+

(+

[*

(*

C *

Ć*

C t

(*

C*

(*

(+

(+

C *

(*

(*

C++

0001

0002

0008

0009

0010

0011

0012 0013

0014

0015

0016

0017

0018

0019

0020

0021

0022

0023

0024

0025

0031 0032 0033

0034 0035

0036 0037

0038 0039 0040

0041

'V04-000'

COPYRIGHT (c) 1978, 1980, 1982, 1984 BY DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. ALL RIGHTS RESERVED.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

AUTHOR BRIAN PORTER

CREATION DATE 9-APR-1979

Functional description:

This module displays entries for the RX211 controller. The format of the error log buffer is as follows.

control and status register
device data buffer register
data path number
data path register
final map register
previous map register
special error register
extended error information

RXD

ARR

2

LAB

FUN

T

COM

, , , ,

```
VAX-11 FORTRAN V3.4-56
DISK$VMSMASTER:[ERF.SRC]RXDISK.FOR;1
                                                         Page
```

```
0058
                   Modified by:
0059
0060
                   V03-003 SAR0233
                                               Sharon A. Reynolds.
                                                                           28-Mar-1984
0061
                            Changed the call to UCB$L_OWNUIC to ORB$L_OWNER.
0062
0063
                   V03-002 SAR0095
                                               Sharon A. Reynolds,
                                                                           20-Jun-1983
0064
                            Changed the carriage control in the 'format' statements
0065
                            for use with ERF.
0066
0067
                   V03-001 SAR0051
                                              Sharon A. Reynolds,
                                                                           13-Jun-1983
                            Removed brief/cryptic support.
8000
0069
0070
                   v02-004 BP0004
                                              Brian Porter,
                                                                           23-NOV-1981
0071
                            Minor edit.
0072
                   v02-003 BP0003
                                              Brian Porter,
                                                                           22-0CT-1981
0074
                                                                    Added 'device attention'
                            Added extended status processing.
0075
                            support.
0076
0077
                   v02-002 BP0002
                                              Brian Porter,
                                                                           23-JUL-1981
0078
                            Added Jifferent uba handling.
0079
0800
                   v02-001 BP0001
                                                                          1-JUL-1981
                                              Brian Porter
0081
                            Added call to DHEAD and LOGGER.
0082
0083
         (--
         [**
0084
0085
                   INCLUDE 'SRC$: MSGHDR. FOR /NOLIST'
0144
                   INCLUDE 'SRC$: DEVERR. FOR /NOLIST'
0245
0246
0247
0248
                  byte
                                     lun
                                     RX2_CS
RX2_ES
UBA_REGS(4)
0249
                   INTEGER+4
0250
0251
                   INTEGER+4
                   INTEGER+4
0252
                   integer*4
                                     special_error_register
0253
                   integer*4
                                     extended_registers(2)
0254
0255
                   integer*2
                                     extended_status(4)
0256
0257
                                     (RX2_CS,EMB$L_DV_REGSAV(0))
(RX2_ES,EMB$L_DV_REGSAV(1))
(UBA_REGS,EMB$L_DV_REGSAV(2))
                   EQUIVALENCE
0258
                   EQUIVALENCE
0259
                   EQUIVALENCE
0260
                                     (special_error_register.emb$l_dv_regsav(6))
(extended_registers.emb$l_dv_regsav(7))
                   equivalence
0261
                   equivalence
0262
                   equivalence
                                      (extended_registers,extended_status)
0263
0264
                   INTEGER+4
                                     FIELD
                                     COMPRESSO
0265
                   INTEGER+4
0266
                   INTEGER+4
                                     COMPRESS4
0267
                   integer * 4
                                     definitive_error_code
0268
                   integer*4
                                     word_count_register
0269
                   integer*4
                                     current_track_address_driveO
0270
                                     current[track[address[drive1
target_track
                   integer*4
0271
                   integer*4
0272
                                     target_sector
                   integer*4
```

L 14

16-Sep-1984 00:14:14 5-Sep-1984 14:21:51

COP

RXC

D

Page

VAX-11 FORTRAN V3.4-56

DISK\$VMSMASTER: [ERF.SRC]RXDISK.FOR: 1

```
0273
0274
                     integer*4
                                         soft_status
                     integer*4
                                         bad_track
0275
0276
                     logical*1
                                         done
0277
0278
                     CHARACTER+20
                                         RX_HEADSELCT(0:1)
0279
                     CHARACTER+7
                                         RX_DENSITY(0:1)
0280
0281
                     CHARACTER*5
                                         DRIVE_TYPE(0:1)
0282
                               DRIVE TYPE(0) /'RX01*'/
                     DATA
0283
                               DRIVE_TYPE(1)
                                                   /'RX02*'/
                     DATA
0284
0285
                     CHARACTER*17
                                        V1RX2_CS(5:7)
                               V1RX2_CS(5)
V1RX2_CS(6)
V1RX2_CS(7)
0286
                     DATA
                                                   /'DONE * '/
0287
                                                   /'INTERRUPT ENABLE*'/
                     DATA
0288
                                                   /'TRANSFER REQUEST*'/
                    DATA
0289
                              er*20 v2rx2_cs(8:9,0:1)
v2rx2_cs(8,0) /'SINGLE U
v2rx2_cs(8,1) /'DOUBLE U
v2rx2_cs(9,0) /'LOWER HE
v2rx2_cs(9,1) /'UPPER HE
0290
                     character*20
0291
                                                   /'SINGLE DENSITY*'/
                     data
0292
                     data
                                                   /'DOUBLE DENSITY+'/
0293
                                                   /'LOWER HEAD SELECTED+'/
                     data
0294
                                                  /'UPPER HEAD SELECTED+'/
                     data
0295
0296
                     CHARACTER*6
                                         V3RX2_CS(15:15)
0297
                               v3rx2_cs(15)
                                                  /'ERROR*'/
                    data
0298
                    CHARACTER*20 V1R)
DATA V1RX2_ES(0)
DATA V1RX2_ES(1)
DATA V1RX2_ES(2)
DATA V1RX2_ES(3)
DATA V1RX2_ES(4)
0299
                                        V1RX2_ES(0:4)
0300
                                                   /'CRC ERROR*'/
0301
                                                   /'SIDE 1 READY (RX03)+'/
0302
                                                   /'INITIALIZE DONE+'/
0303
                                                   /'DRIVE AC LO+'/
0304
                                                   /'DENSITY ERROR+'/
0305
                              r*21 v2rx2_es(5:5,0:1)
v2rx2_es(5,0) /'SINGLE DENSITY DRIVE*'/
v2rx2_es(5,1) /'DOUBLE DENSITY DRIVE*'/
0306
                    character*21
0307
                    data
0308
                    data
0309
0310
                    CHARACTER*13
                                      V3RX2_ES(6:7)
                               v3rx2_es(6)
v3rx2_es(7)
0311
                                                  /'DELETED DATA+'/
                    data
                                                   /'DRIVE READY*'/
0312
                     data
0313
0314
                                      V4RX2_ES(10:11)
                    CHARACTER+20
                               v4rx2_es(10)
v4rx2_es(11)
0315
                                                  /'WORD COUNT OVERFLOW+'/
                    data
0316
                                                   /'NON-EXISTENT MEMORY+'/
                     data
0317
0318
                               r*27 v1soft_status(0:0,0:1)
v1soft_status(0,0) /'COMMAND WAS SINGLE DENSITY*'/
                    character*27
0319
                     data
0320
                               v1soft_status(0,1)
                                                             /'COMMAND WAS DOUBLE DENSITY*'/
                     data
0321
                    character*26 v2soft_status(4:4,0:1)
data v2soft_status(4.0) /DRIVE #0., SINGLE DENSITY*'/
0322
0323
0324
                               v2soft_status(4,1)
                                                              /'DRIVE #O., DOUBLE DENSITY*'/
                     data
0325
0326
0327
0328
                     character*12
                                       v3soft_status(5:5)
                               v3soft_status(5)
                                                             /'HEAD LOADED*'/
                     data
0329
                     character*26
                                         v4soft_status(6:7,0:1)
```

```
RXDISK
```

```
16-Sep-1984 00:14:14
5-Sep-1984 14:21:51
0330
0331
0332
0333
                                                                  /'DRIVE #1., SINGLE DENSITY*'/
/'DRIVE #1., DOUBLE DENSITY*'/
/'DRIVE #0. SELECTED*'/
                                  v4soft_status(6,0)
                      data
                                 v4soft_status(6,1)
v4soft_status(7,0)
                      data
                      data
                                 v4soft_status(7,1)
                                                                   /'DRIVE #1. SELECTED+'/
                      data
0334
0335
0336
0337
0338
                      CALL FRCTOF (LUN)
                      call dhead1 (lun, 'UBA RX211')
                      done = .false.
0340
0341
                      if (lib$extzv (5,1,rx2_cs) .eq. 1) done = .true.
0342
                      CALL LINCHK (LUN,2)
0344
0345
0346
0347
0348
0350
0351
                      WRITE(LUN, 20) RX2_CS
FORMAT(/' ', T8, 'RX2CS', T24, Z8.4)
           20
                      if (done) then
                      call linchk (lun,1)
0352
0353
                      write(lun,25) lib$extzv(4,1,rx2_cs)
format(' ',t40,'DRIVE #',i1,'. SELECTED')
           25
0354
                      endif
0355
0356
                      CALL OUTPUT (LUN,RX2_CS,V1RX2_CS,5,5,7,'0')
0357
0358
                      if (done) then
0359
0360
                      call output (lun,rx2_cs,v2rx2_cs,8,8,9,'2')
0361
                      endif
0362
0363
                      FIELD=LIBSEXTZV(11,1,RX2_CS)
0364
0365
                      CALL LINCHK (LUN, 1)
0366
                      WRITE(LUN,45) DRIVE_TYPE(FIELD)
FORMAT(' ',T40,'CONTROLLER DRIVE TYPE ',
1 A<COMPRESSC (DRIVE_TYPE(FIELD))>)
0367
0368
           45
0369
0370
0371
                      CALL OUTPUT (LUN, RX2_CS, V3RX2_CS, 15, 15, 15, 10')
0372
0373
                      CALL LINCHK (LUN, 1)
0374
0375
                      WRITE(LUN,55) RX2_ES FORMAT(' ', T8, 'RX2ES', T24, Z8.4)
0376
0377
           55
0378
                      if (done) then
0379
0380
                      if (lib$extzv(1,3,rx2_cs) .eq. 7) then
0381
0382
                      call linchk (lun,1)
0383
0384
                      definitive_error_code = lib$extzv(0,7,rx2_es)
0385
0386
```

if (definitive_error_code .eq. '010'o) then

N 14

VAX-11 FORTRAN V3.4-56

DISK\$VMSMASTER:[ERF.SRC]RXDISK.FOR:1

Page

```
write(lun,60)
format(' ',t40,'DRIVE #0., 'HOME' FAILURE ON INIT')
          60
                    else if (definitive error code .eg. '020'o) then
                    write(lun,65)
format(' ',t40,'DRIVE #1., 'HOME' FAILURE ON INIT')
          65
                    else if (definitive_error_code .eq. '040'o) then
                    write(lun,70)
format(' ',t40,'TRIED TO ACCESS TRACK >77.')
          70
                    else if (definitive_error_code .eq. '050'o) then
                    write(lun,75)
format(' ',t40,'''HOME'' BEFORE DESIRED TRACK')
          75
                    else if (definitive_error_code .eq. '070'o) then
                    write(lun,80)
format(' ',t40,'2. REVOLUTIONS, SECTOR NOT FOUND')
          80
                    else if (definitive_error_code .eq. '110'o) then
                    write(lun,85)
format(' ',t40,'>40 MICRO-SEC, NO ''SEP'' CLOCK')
          85
                    else if (definitive_error_code .eq. '120'o) then
                    write(lun,90)
format(' ',t40,'A PREAMBLE NOT FOUND')
0419
0420
0421
0423
0423
0423
          90
                    else if (definitive_error_code .eq. '130'o) then
                    write(lun,95) format(' ',t40,'PREAMBLE BUT NO ''ID''')
          95
                    else if (definitive_error_code .eq. '140'o) then
0427
0428
0429
0430
0431
0433
0433
0438
0438
0441
                    write(lun,100)
format(' ',t40,'CRC ERROR, HEADER')
          100
                    else if (definitive_error_code .eq. '150'o) then
                    write(lun,105)
format(' ',t40,'HEADER/DESIRED TRACK MISMATCH')
          105
                    else if (definitive_error_code .eq. '160'o) then
                    write(lun,110)
format(' ',t40,'TOO MANY TRIES FOR ''IDAM''')
          110
                    else if (definitive_error_code .eq. '200'o) then
0442
                    write(lun,115)
```

C 15

```
0444
         115
                   format(' '.t40.'CRC ERROR, DATA')
0445
0446
                  else if (definitive_error_code .eq. '220'o) then
0447
0448
0449
0450
0451
                  write(lun,120)
format(' ',t40,'DIAGNOSTIC MODE TEST FAILURE')
         120
                  else if (definitive_error_code .eq. '240'n) then
0453
0453
0455
0456
0456
0457
                  write(lun,125)
format(' ',t40,'DENSITY ERROR')
         125
                  else if (definitive_error_code .eq. '250'o) then
                   write(lun,130)
         130
                              ,t40, 'SET DENSITY, INCORRECT 'KEYWORD'")
                   format('
0460
                  endif
0461
                  else
0462
                  CALL OUTPUT (LUN, RX2_ES, V1RX2_ES, 0.0,4,'0')
0464
0465
                  call output (lun,rx2_es,v2rx2_es,5,5,5,'2')
0466
                  CALL OUTPUT (LUN, RX2_ES, V3RX2_ES, 6, 6, 6, 6, 10')
0468
                  endif
2469
0470
                  call output (lun,rx2_es,v3rx2_es,6,7,7,'0')
0471
0472
                  CALL LINCHK (LUN,1)
0473
0474
                  write(lun,25) libSextzv (8,1,rx2_es)
0475
0476
                  CALL OUTPUT (LUN, RX2_ES, V3RX2_ES, 10, 10, 11, '0')
0477
                  endif
0478
0479
                  call linchk (lun.4)
0480
                  write(lun,135) (extended_status(i),i = 1,4)
format(' ',t8,'EXTENDED STATUS',4(t28,z4.4,:/))
0481
0482
0483
         135
0484
                  definitive_error_code = libSextzv(0,8,extended_registers(1))
0485
0486
                  call linchk (lun.1)
0487
0488
                  if (definitive_error_code .eq. '010'o) then
0489
0490
                  write(lun,60)
0491
0492
                  else if (definitive_error_code .eq. '020'o) then
0494
                  write(lun,65)
0495
0495
                  else if (definitive_error_code .eq. '040'o) then
0497
0498
                  write(lun,70)
0499
                  else if (definitive_error_code .eq. '050'o) then
```

0501 0502 0503

0504

0516

0518

0538

0539 0540

0541

0548 0549

0550 0551

0552

0554

0555 0556

0557

```
write(lun.75)
        else if (definitive_error_code .eq. '070'o) then
        write(lun,80)
        else if (definitive_error_code .eq. '110'o) then
        write(lun.85)
        else if (definitive_error_code .eq. '120'o) then
        write(lun,90)
        else if (definitive_error_code .eq. '130'o) then
        write(lun,95)
        else if (definitive_error_code .eq. '140'o) then
        write(lun,100)
        else if (definitive_error_code .eq. '150'o) then
        write(lun,105)
        else if (definitive_error_code .eq. '160'o) then
        write(lun,110)
        else if (definitive_error_code .eq. '200'o) then
        write(lun,115)
        else if (definitive_error_code .eq. '220'o) then
        write(lun,120)
        else if (definitive_error_code .eq. '240'o) then
        write(lun,125)
        else if (definitive_error_code .eq. '250'o) then
        write(lun,130)
        endif
        word_count_register = libSextzv(8,8,extended_registers(1))
        call linchk (lun,1)
        write(lun,137) 'WORD COUNT REGISTER ',word_count_register,' (HEX)'
format(' ',t40,a,z4.4,a)
137
        current_track_address_drive0 = lib$extzv(16,8,extended_registers(1))
```

```
0558
0559
0560
0561
0562
0563
                  call linchk (lun.1)
                  write(lun,140) current_track_address_drive0
format(' ',t40,'CURRENT_TRACK #',
         140
                  1 i<compress4 (current_track_address_drive0)>,'., DRIVE #0.')
0564
0565
                  current_track_address_drive1 = libSextzv(24,8,extended_registers(1))
0566
0567
                  call linchk (lun,1)
                  write(lun,145) current_track_address_drive1
format(' ',t40,'CURRENT TRACK #',
0568
0569
0570
         145
                   1 i<compress4 (current_track_address_drive1)>,'., DRIVE #1.')
0571
0572
                   target_track = libSextzv(0,8,extended_registers(2))
0573
0574
                   call linchk (lun,1)
0575
0576
                   write (lun,150) target_track
                  format(' ',t40,'TRACK ",i<compress4 (target_track)>,
0577
         150
0578
                   1 '.. TARGÉT TRACK')
0579
0580
                   target_sector = lib$extzv(8,8,extended_registers(2))
0581
0582
                  call linchk (lun,1)
0583
                  write(lun,155) target_sector
format(' ',t40,'SECTOR #',i<compress4 (target_sector)>,
0584
0585
         155
0586
                  1 '.. TARGÉT SÉCTOR')
0587
0588
                  soft_status = lib$extzv(16,8,extended_registers(2))
0589
0590
                  call output (lun, soft_status, v1soft_status, 0, 0, 0, '2')
0591
0592
                  call output (lun, soft_status, v2soft_status, 4,4,4,'2')
0593
0594
                  call output (lun, soft_status, v3soft_status, 5,5,5,'0')
0595
0596
                  call output (lun, soft_status, v4soft_status, 6,6,7,'2')
0597
0598
                  if (definitive_error_code .eq. '150'o) then
0599
0600
                  bad_track = lib$extzv(24,8,extended_registers(2))
0601
0602
                  call linchk (lun,1)
0603
                  write(lun,160) bad track format(' ',t40,'SELECTED DRIVE AT TRACK #',
0604
0605
         160
0606
                   1 i<compress4 (bad_track)>,'.')
0607
                  endif
0608
0609
                   if (emb$w_hd_entry .ne. 98) then
0610
0611
                  call uba_datapath (lun,uba_regs(1),uba_regs(2))
0612
0613
                  call uba_mapping (lun,-1,uba_regs(3))
0614
```

SB

VAX-11 FORTRAN V3.4-56 DISKSVMSMASTER:[ERF.SRC]RXDISK.FOR;1

```
RXDISK
                                                                         16-Sep-1984 00:14:14
5-Sep-1984 14:21:51
0615
0616
0617
                  if (libSextzv (16,16,emb$l_dv_iosb1) .qt. 512) then
                  call uba_mapping (lun,-1,uba_regs(4))
0618
                  endif
9012334567890123345678906633367890
                  call linchk (lun,1)
                  write(lun,165)
format(' ,:)
         165
                  call ucb$b_ertcnt (lun,emb$b_dv_ertcnt)
                  call_ucb$b_ertmax (lun,emb$b_dv_ertmax)
                  endif
                  call orb$l_owner (lun,emb$l_dv_ownuic)
                  call ucb$l_char (lun,emb$l_dv_char)
                  call ucb$w_sts (lun,emb$w_dv_sts)
                  call ucb$l_opcnt (lun,emb$l_dv_opcnt)
                  call ucb$w_errcnt (lun,emb$w_dv_errcnt)
                  if (emb$w_hd_entry .ne. 98) then
0641
0642
                  call ucb$l_media (lun,emb$l_dv_media)
0644
                  call linchk (lun,1)
0645
0646
                  write(lun.165)
0647
0648
                  call rxdisk_qio (lun,emb$w_dv_func)
0649
0650
                  call irp$w_bcnt (lun,emb$w_dv_bcnt)
0651
0652
0653
                  call irp$w_boff (lun,emb$w_dv_boff)
0654
                  call irp$l_pid (lun,emb$l_dv_rqpid)
0655
0656
0657
                  call irp$q_iosb (lun,emb$l_dv_iosb1)
                  endif
0658
0659
                  RETURN
0660
                  END
```

F 15

VAX-11 FORTRAN V3.4-56 Page 10 DISK\$VMSMASTER:[ERF.SRC]RXDISK.FOR;1

PROGRAM SECTIONS

Name	Bytes	Attributes
O SCODE 1 SPDATA 2 SLOCAL 3 EMB	NTA 962	PIC CON REL LCL SHR EXE RD NOWRT LONG PIC CON REL LCL SHR NOEXE RD NOWRT LONG PIC CON REL LCL NOSHR NOEXE RD WRT LONG PIC OVR REL GBL SHR NOEXE RD WRT LONG
Total Space Allocated	6123	

ENTRY POINTS

```
Address Type Name
0-00000000 RXDISK
```

VARIABLES

Address	Type	Name	Address	Type	Name
2-0000298 2-0000288 2-0000277 3-0000010 3-000001D 3-0000012 3-0000026 3-000002E 3-000003F 3-000003F 3-000003C 3-000002A 3-000002A 3-0000029C 3-000029C 3-0000290 2-0000290 2-0000280	1*4 1*1 1*1 1*4 1*4 1*4 1*4 1*4 1*4 1*4	BAD_TRACK CURRENT_TRACK_ADDRESS_DRIVE1 DONE EMB\$B_DV_ERTCNT EMB\$B_DV_NAMLNG EMB\$B_DV_TYPE EMB\$L_DV_IOSB1 EMB\$L_DV_MEDIA EMB\$L_DV_ROPID EMB\$L_DV_ROPID EMB\$T_DV_NAME EMB\$W_DV_BOFF EMB\$W_DV_FUNC EMB\$W_DV_UNIT EMB\$W_HD_ERRSEQ I RX2_CS SOFT_STATUS TARGET_SECTOR WORD_COUNT_REGISTER	2-0000284 2-000027C 3-000001C 3-000003A 3-0000036 3-0000016 3-0000004E 3-0000002C 3-000002C 3-000002C 3-000001A 3-000001A 3-000002C 3-00000278 AP-00000056 3-0000056		CURRENT TRACK ADDRESS_DRIVEO DEFINITIVE_ERROR_CODE EMB\$B_DV_CLASS EMB\$B_DV_ERTMAX EMB\$B_DV_SLAVE EMB\$L_DV_CHAR EMB\$L_DV_IOSB2 EMB\$L_DV_NUMREG EMB\$L_DV_OWNUIC EMB\$L_HD_SID EMB\$W_DV_BCNT EMB\$W_DV_ERRCNT EMB\$W_DV_STS EMB\$W_DV_STS EMB\$W_HD_ENTRY FIELD LUN RX2_ES SPECIAL_ERROR_REGISTER TARGET_TRACK

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-0000036 3-0000000 3-00000052 3-0000006 3-0000006E 3-0000006E 2-00000000	L+1 I+4 I+4 I+2 CHAR	DRIVE_TYPE EMB EMB\$L_DV_REGSAV EMB\$Q_HD_TIME EXTENDED_REGISTERS EXTENDED_STATUS RX_DENSITY RX_HEADSELCT	10 512 420 8 8 8 14 40	(0:1) (0:511) (0:104) (2) (2) (4) (0:1) (0:1)

SB

RXDISK	н 15 16-Sep-1984 00:14:14 5-Sep-1984 14:21:51	VAX-11 FORTRAN V3.4-56 Page 11 DISK\$VMSMASTER:[ERF.SRC]RXDISK.FOR;1			
3-0000005A I+4 UBA_REGS 2-00000040 CHAR V1RX2_CS 2-00000009 CHAR V1RX2_ES 2-00000199 CHAR V1SOFT_STATUS 2-0000012D CHAR V2RX2_ES 2-000001CF CHAR V2RX2_ES 2-000001CF CHAR V3RX2_ES 2-00000157 CHAR V3RX2_ES 2-00000157 CHAR V3RX2_ES 2-00000203 CHAR V3SOFT_STATUS 2-00000203 CHAR V4RX2_ES 2-00000205 CHAR V4RX2_ES	16 (4) 51 (5:7) 100 (0:4) 54 (0:0, 0:1) 80 (8:9, 0:1) 42 (5:5, 0:1) 52 (4:4, 0:1) 6 (15:15) 26 (6:7) 12 (5:5) 40 (10:11) 104 (6:7, 0:1)				
LABELS					
Address Label Address Label	Address Label Address Label	Address Label Address Label			
1-00000068 20' 1-0000007B 25' 1-00000120 70' 1-00000142 75'	1-00000098 45' 1-000000BC 55' 1-0000018D 85'	1-000000CE 60' 1-000000F7 65' 1-000001B2 90' 1-000001CE 95'			
1-000001EA 100' 1-00000203 105' 1-00000299 130' 1-000002C1 135' 1-0000036B 155' 1-00000393 160'	1-00000228 110' 1-00000249 115' 1-000002E4 137' 1-000002EF 140' 1-000003BD 165'	1-00000260 120' 1-00000284 125' 1-0000031A 145' 1-00000345 150'			
FUNCTIONS AND SUBROUTINES REFERENCED					
Type Name Type Name	Type Name Type Name	Type Name Type Name			
I+4 COMPRESS4 I+4 COMPRESSC IRP\$W_BCNT IRP\$W_BOFF RXDISK_QIO UBA_DATAPATH UCB\$L_MEDIA UCB\$L_OPCNT	DHEAD1 FRCTOF I*4 LIBSEXTZV LINCHK UBA_MAPPING UCB\$B_ERTCNT UCB\$W_ERRCNT UCB\$W_STS	IRP\$L_PID IRP\$Q_IOSB ORB\$L_OWNER OUTPUT UCB\$B_ERTMAX UCB\$L_CHAR			

Page 12

```
Subroutine RXDISK_QIO (lun,emb$w_dv_func)
include 'src$:giocommon.for /nolist'
byte
                Lun
integer*2
                emb$w_dv_func
integer*4
                qiocode(0:1,0:63)
if (qiocode(0,0) .eq. 0) then
qiocode(1,08) = %loc(io$_packack)
qiocode(1,11) = %loc(io$_writepblk)
qiocode(1,12) = %loc(io%_readpblk)
qiocode(1,26) = %loc(io$_setchar)
qiocode(1,27) = %loc(io%_sensechar)
qiocode(1,30) = %loc(io%_format)
```

0001 0002 0003

0004

0005

0300

0304 0305

0306

0308 0309

0310

0314 0315

0316 0317

0318 0319 0320

qiocode(1,32) = %loc(io\$_writelblk) qiocode(1,33) = %loc(io%_readlblk) qiocode(1,35) = %loc(io%_setmode) qiocode(1,39) = %loc(io%_sensemode) qiocode(1,48) = %loc(io\$_writevblk) qiocode(1,49) = %loc(io%_readvblk) qiocode(1,50) = %loc(io\$_access) qiocode(1,51) = %loc(io%_create) qiocode(1,52) = %loc(io%_deaccess) qiocode(1,53) = %loc(io\$_delete) qiocode(1,54) = %loc(ios_modify) qiocode(1,56) = %loc(io%_acpcontrol) qiocode(1,57) = %loc(io%_mount) do 10, i = 0.63

```
J 15
RXDISK_QIO
                                                                                                                     16-Sep-1984 00:14:14
                                                                                                                                                                VAX-11 FORTRAN V3.4-56
                                                                                                                                                                                                                                  Page 13
                                                                                                                       5-Sep-1984 14:21:51
                                                                                                                                                                DISKSVMSMASTER: [ERF.SRC]RXDISK.FOR: 1
aiocode(0.i) = 33
                             if (giocode(1,i) .eq. 0) then
                             qiocode(1,i) = %loc(qio_string)
                             endif
              10
                             continue
                             endif
                             call irp$w_func (lun,emb$w_dv_func,
                             1 giocode(0,libSextzv(0,6,embSw_dv_func)))
0334
                             return
0335
0336
                             end
PROGRAM SECTIONS
                                                                          Bytes
                                                                                         Attributes
       Name
                                                                                         PIC CON REL LCL
   O SCODE
                                                                               225
                                                                                                                          SHR
                                                                                                                                   EXE
                                                                                                                                                RD NOWRT LONG
    1 SPDATA
                                                                                                                          SHR NOEXE
                                                                                                                                                RD NOWRT LONG
                                                                               548
      $LOCAL
                                                                                         PIC CON REL LCL NOSHR NOEXE
                                                                                                                                                RD
                                                                                                                                                         WRT LONG
   3 QTOCOMMON
                                                                            1247
                                                                                         PIC OVR REL GBL
                                                                                                                          SHR NOEXE
                                                                                                                                                RD
                                                                                                                                                         WRT LONG
       Total Space Allocated
                                                                             2028
ENTRY POINTS
       Address Type Name
   0-00000000
                                  RXDISK_QIO
VARIABLES
       Address Type Name
                                                                                                    Address Type Name
AP-0000008a I+2 EMB$W DV FUNC

3-0000442 CHAR IO$ ABORT

3-00003C2 CHAR IO$ ACPCONTROL

3-0000297 CHAR IO$ CLEAN

3-0000385 CHAR IO$ DEACCESS

3-000026D CHAR IO$ DIAGNOSE

3-000004CB CHAR IO$ DSE

3-00000276 CHAR IO$ FORMAT

3-00000276 CHAR IO$ LOADMCODE

3-000003E2 CHAR IO$ MOUNT

3-000009D CHAR IO$ OFFSET

3-000000E0 CHAR IO$ QSTOP

3-00000421 CHAR IO$ READCSR

3-00000286 CHAR IO$ READCSR
                                                                                                2-00000200
3-0000034D
3-000004B3
3-00000369
3-00000065
3-000000071
3-00000071
3-00000000
                                                                                                                     CHAR IOS_ACCESS
CHAR IOS_AVAILABLE
CHAR IOS_CREATE
CHAR IOS_DELETE
CHAR IOS_DELETE
CHAR IOS_ERASETAPE
CHAR IOS_INITIALIZE
CHAR IOS_MODIFY
CHAR IOS_PACKACK
CHAR IOS_RDSTATS
CHAR IOS_READHEAD
CHAR IOS_READPBLK
                                                                                                                      1+4
                                                                                                  Ĭ-ŇŎŎŎŎŎĔĔ
                                                                                                 3-000003EF
                                                                                                 3-00000169
```

3-0000013f

SB

068

PR(

(

EN'

VAF

```
K 15
                                                                    16-Sep-1984 00:14:14
5-Sep-1984 14:21:51
                                                                                            VAX-11 FORTRAN V3.4-56
DISK$VMSMASTER:[ERF.SRC]RXDISK.FOR;1
RXDISK_Q10
                                                                                                                                     Page 14
                                                        3-00000200 CHAR IOS_READPRESET
3-0000033A CHAR IOS_READVBLK
               CHAR IOS READWITHXBUF
  3-00000484
 3-0000007C
               CHAR IOS RELEASE
 AP-00000004a L+1 LUN
                                                         3-000004A1
                                                                     CHAR QIO_STRING
ARRAYS
                                         Bytes Dimensions
    Address Type Name
  2-00000000 I+4 Q10C0DE
                                       512 (0:1, 0:63)
LABELS
    Address
              Label
               10
FUNCTIONS AND SUBROUTINES REFERENCED
  Type Name
                         Type Name
        IRP$W_FUNC
                        I+4 LIBSEXTZV
COMMAND QUALIFIERS
  FORTRAN /LIS=LISS:RXDISK/OBJ=OBJS:RXDISK MSRCS:RXDISK
  /CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
  /STANDARD=(NOSYNTAX, NOSOURCE_FORM)
/SHOW=(NOPREPROCESSOR, NOINCLUDE, MAP)
  /f77 /NOG_FLOATING /14 /OPTIMIZE /WARNINGS /NOD_LINES /NOCROSS_REFERENCE /NOMACHINE_CODE /CONTINUATIONS=19
```

SB

ARI

LA

FUI

RXDISK_QIO

L 15 16-Sep-1984 00:14:14 5-Sep-1984 14:21:51

VAX-11 FORTRAN V3.4-56 DISK\$VMSMASTER: [ERF.SRC]RXDISK.FOR; 1

Page 15

COMPILATION STATISTICS

9.73 seconds 23.45 seconds 249 235 pages Run Time: Elapsed Time: Page Faults: Dynamic Memory:

0153 AH-BT13A-SE

DIGITAL EQUIPMENT CORPORATION CONFIDENTIAL AND PROPRIETARY

